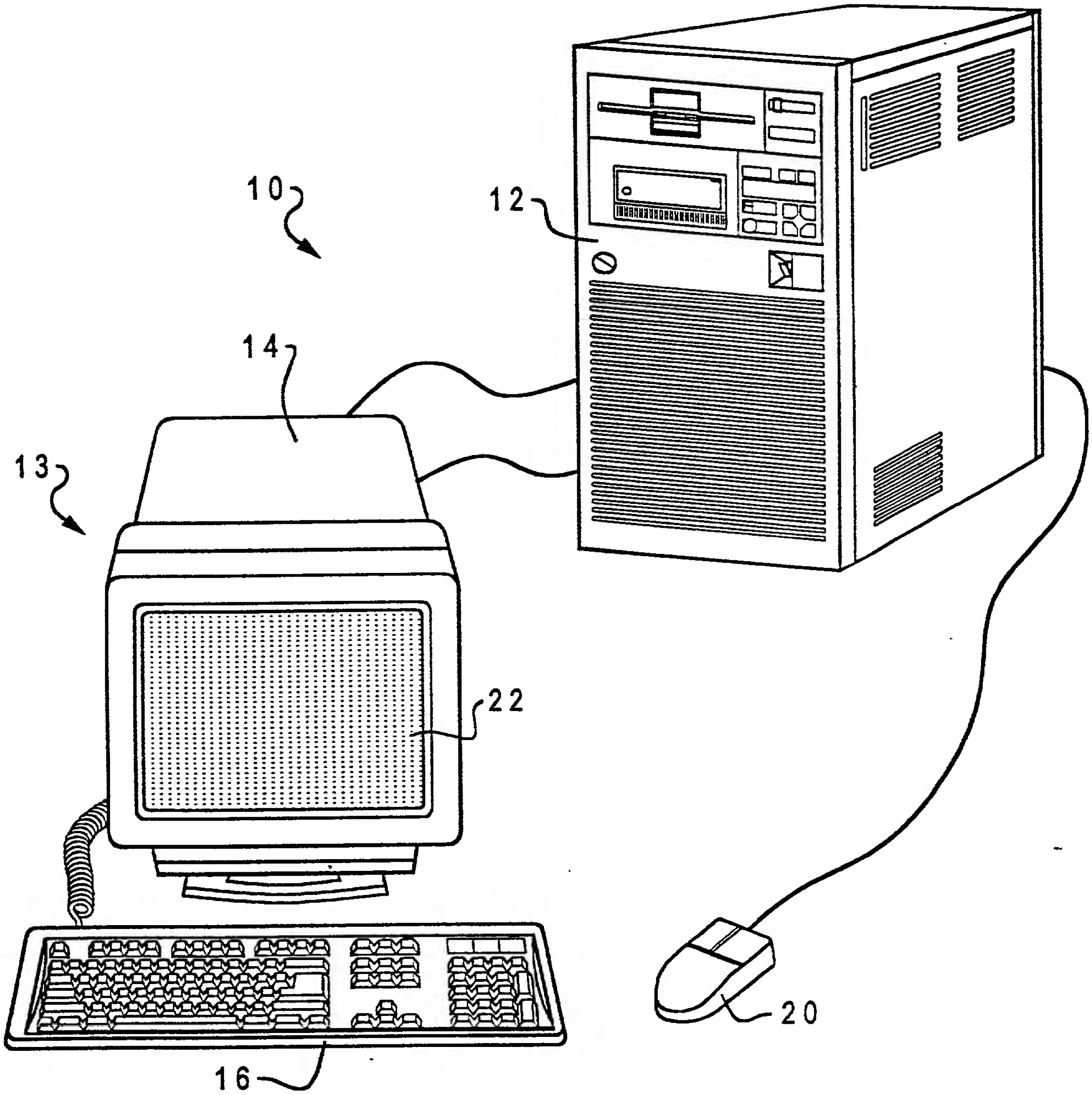


Fig. 1

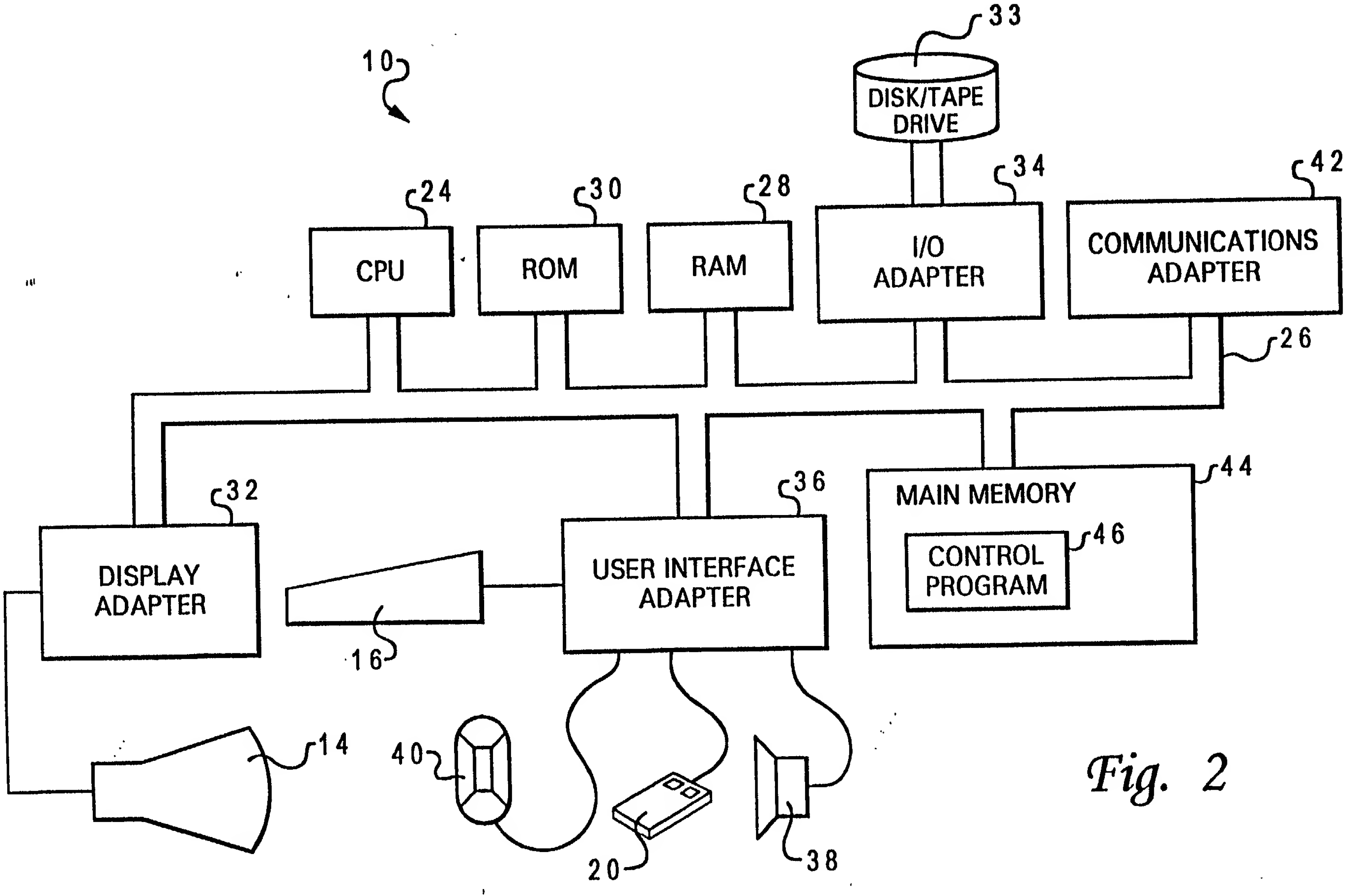
10
33
DISK/TAPE DRIVE
24
CPU
30
ROM
28
RAM
34
I/O ADAPTER
42
COMMUNICATIONS ADAPTER
26
44
MAIN MEMORY
46
CONTROL PROGRAM
32
DISPLAY ADAPTER
16
36
USER INTERFACE ADAPTER
14
40
20
38

Fig. 2

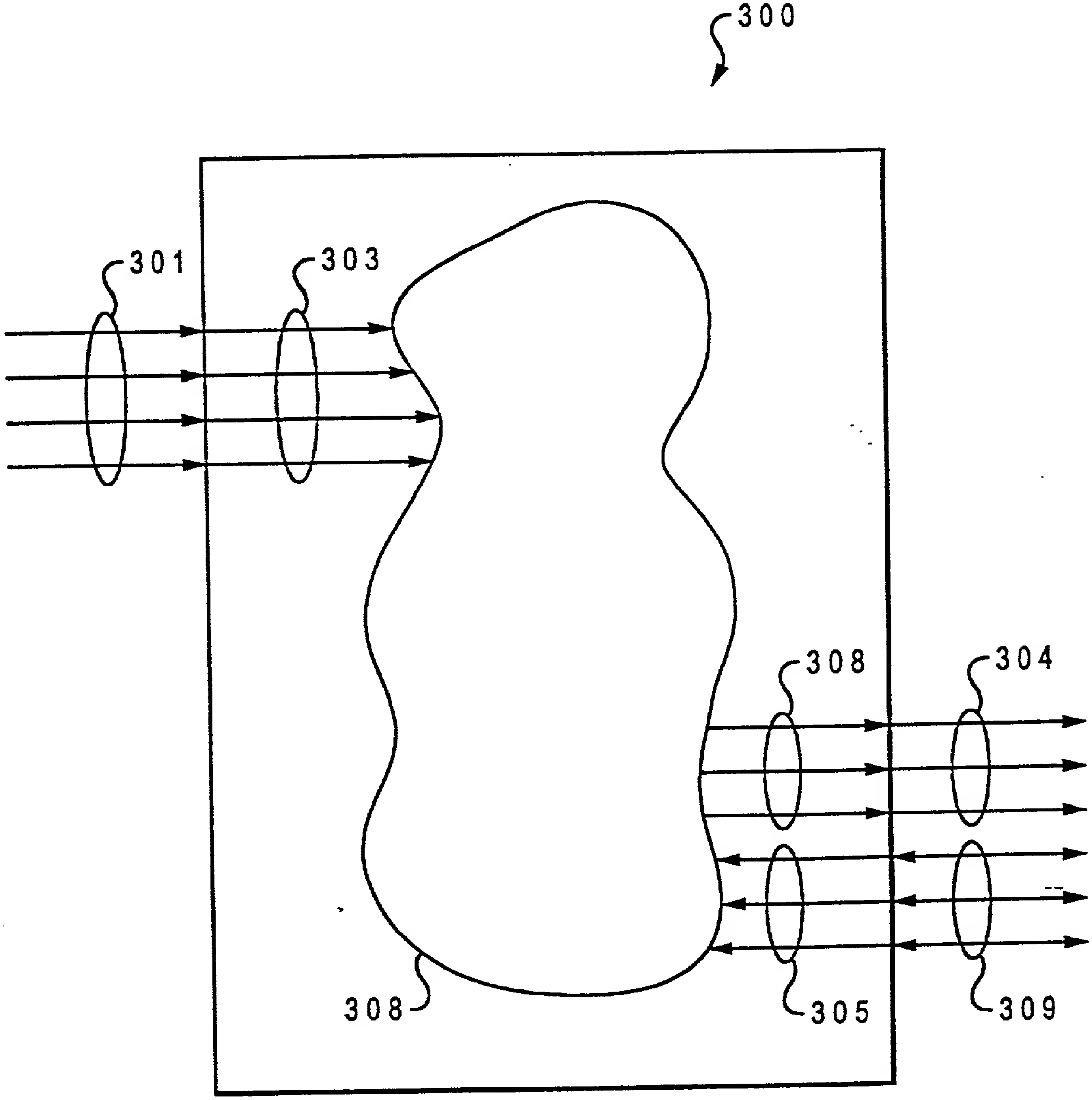


Fig. 3A

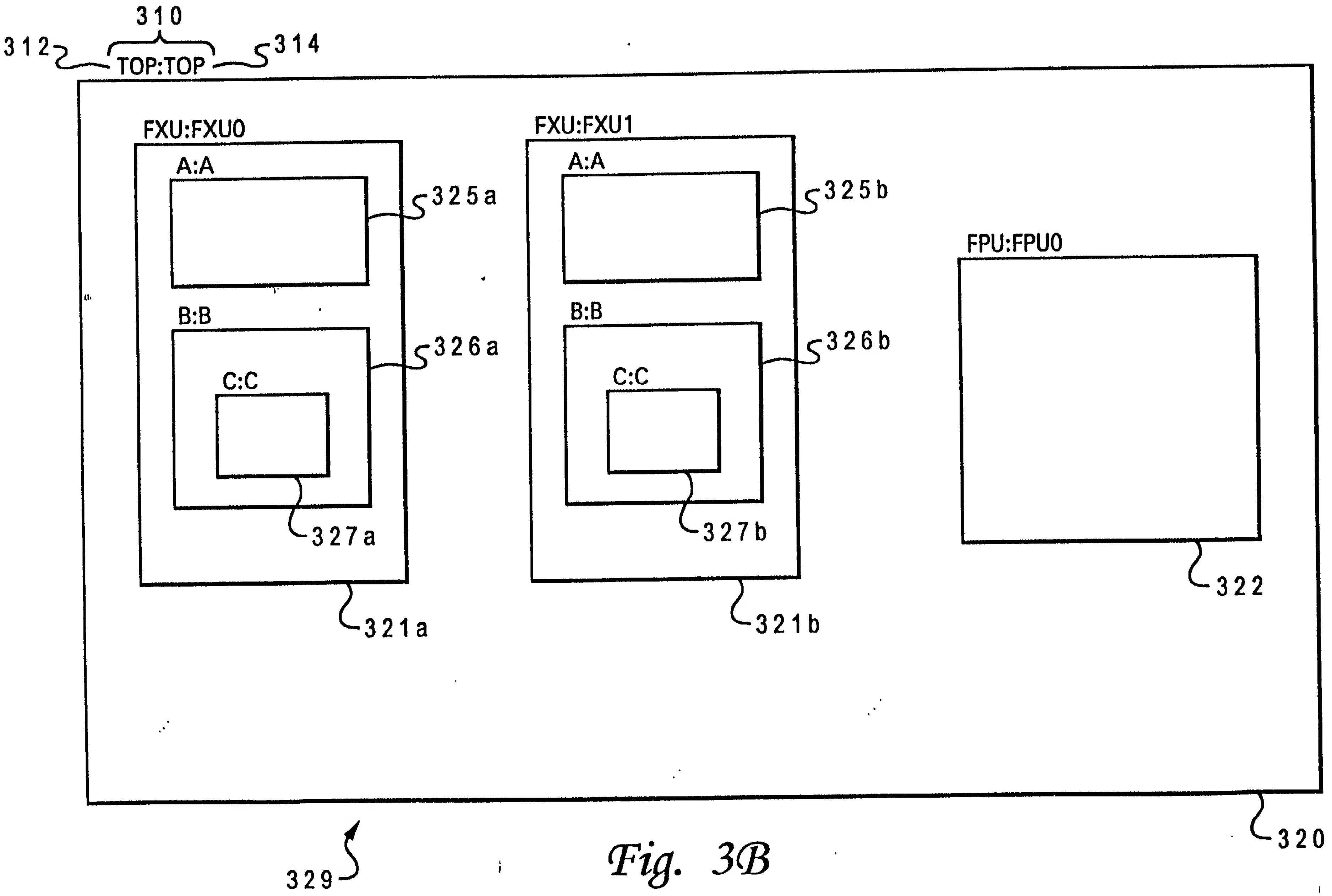

310
312
TOP:TOP
314
FXU:FXU0
A:A
325a
B:B
326a
C:C
327a
321a
FXU:FXU1
A:A
325b
B:B
326b
C:C
327b
321b
FPU:FPU0
322
320
329

Fig. 3B

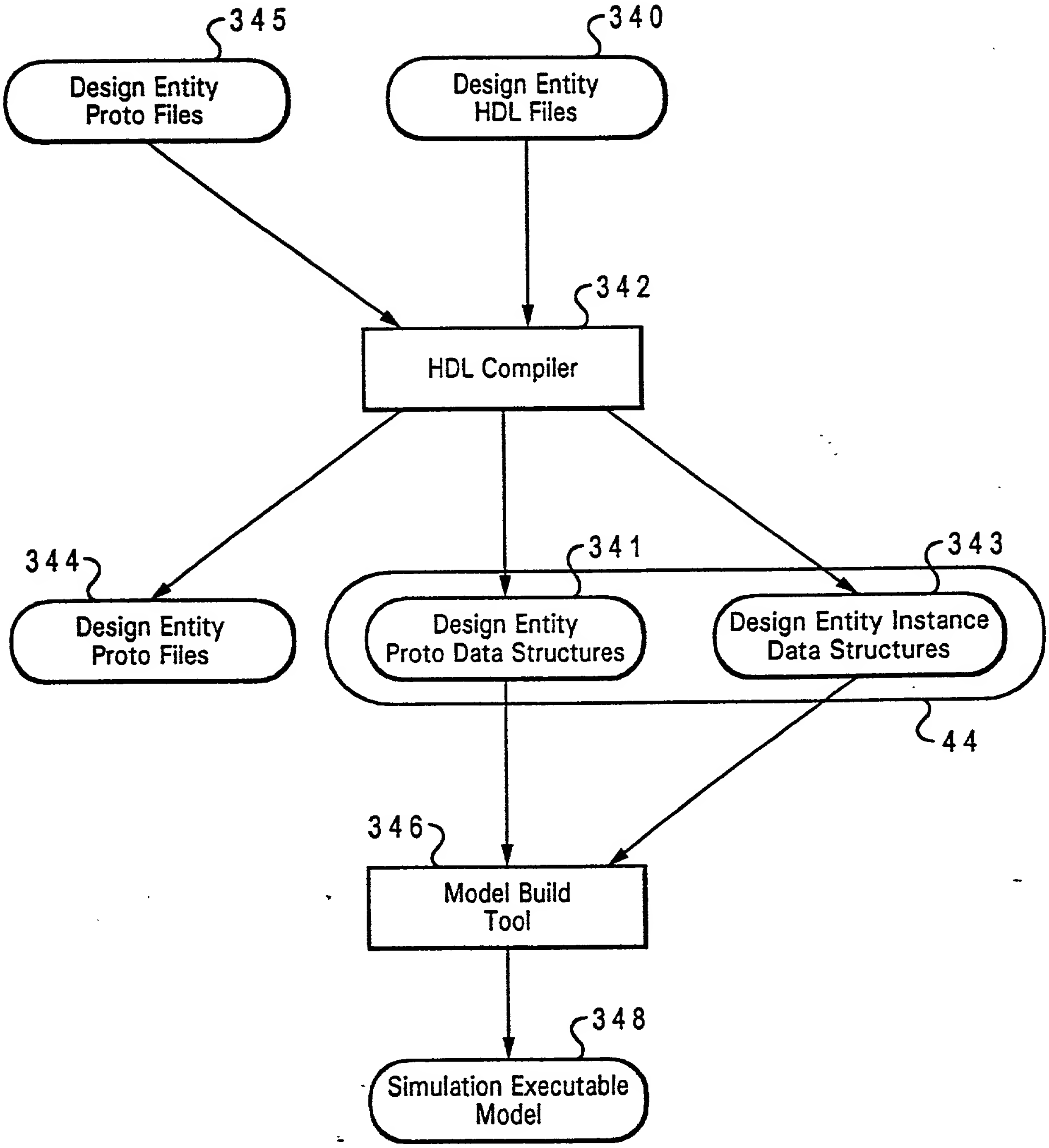

345
Design Entity Proto Files
340
Design Entity HDL Files
342
HDL Compiler
344
Design Entity Proto Files
341
Design Entity Proto Data Structures
343
Design Entity Instance Data Structures
44
346
Model Build Tool
348
Simulation Executable Model

Fig. 3C

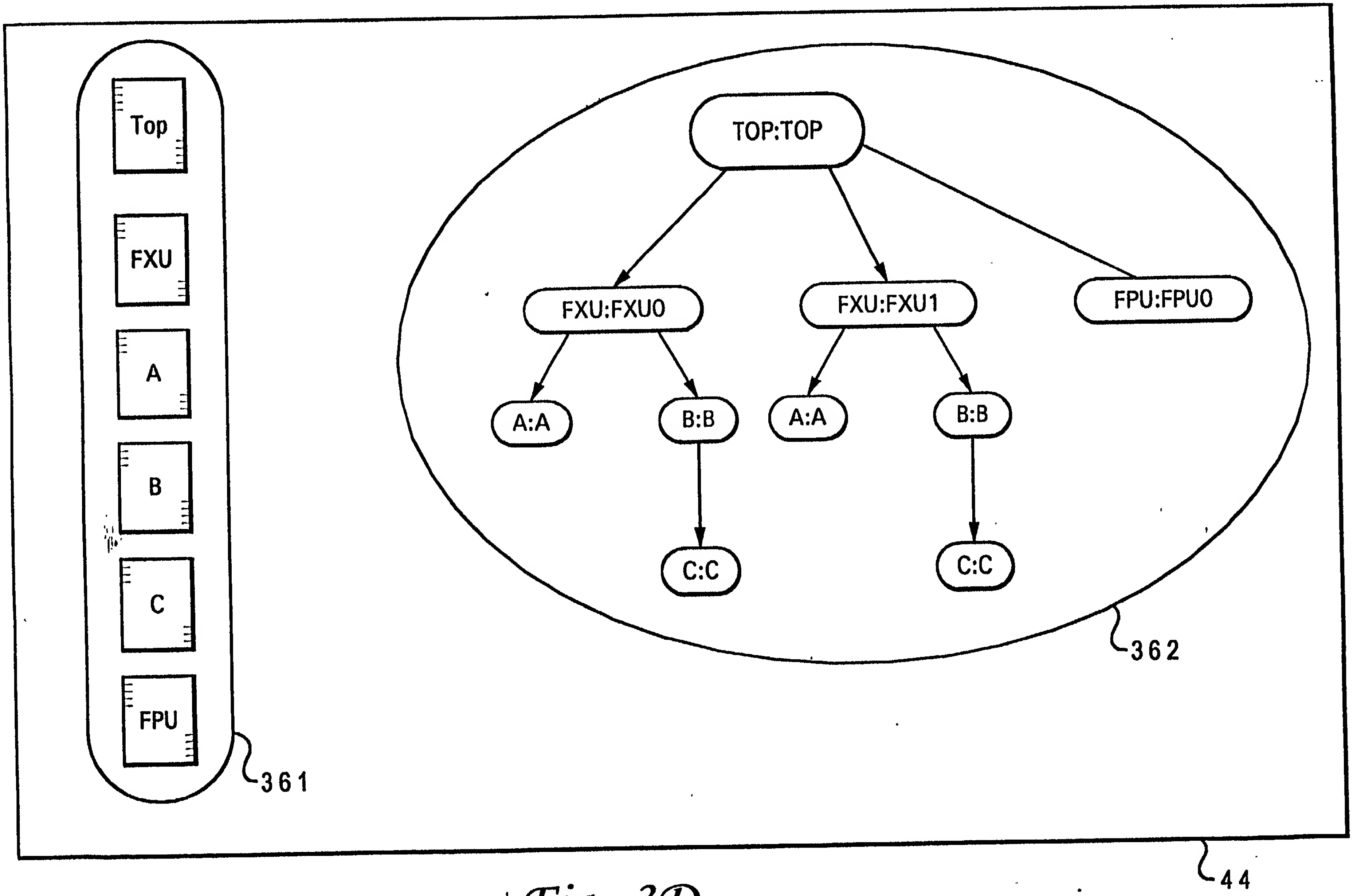

Top
FXU
A
B
C
FPU
361
TOP:TOP
FXU:FXU0
FXU:FXU1
FPU:FPU0
A:A
B:B
A:A
B:B
C:C
C:C
362
44

Fig. 3D

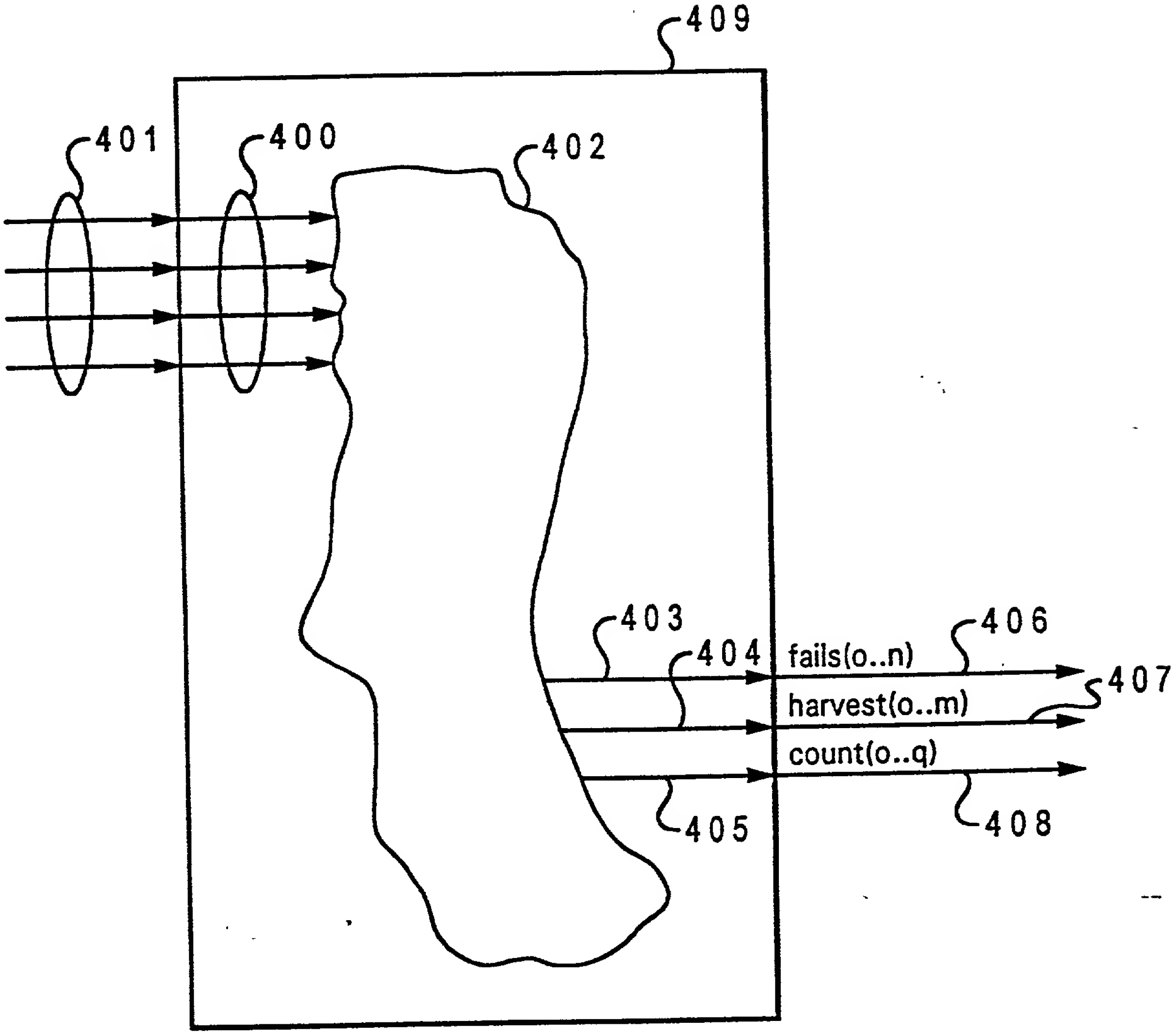

409
401
400
402
403
404
fails(o..n)
406
407
harvest(o..m)
count(o..q)
405
408

Fig. 4A

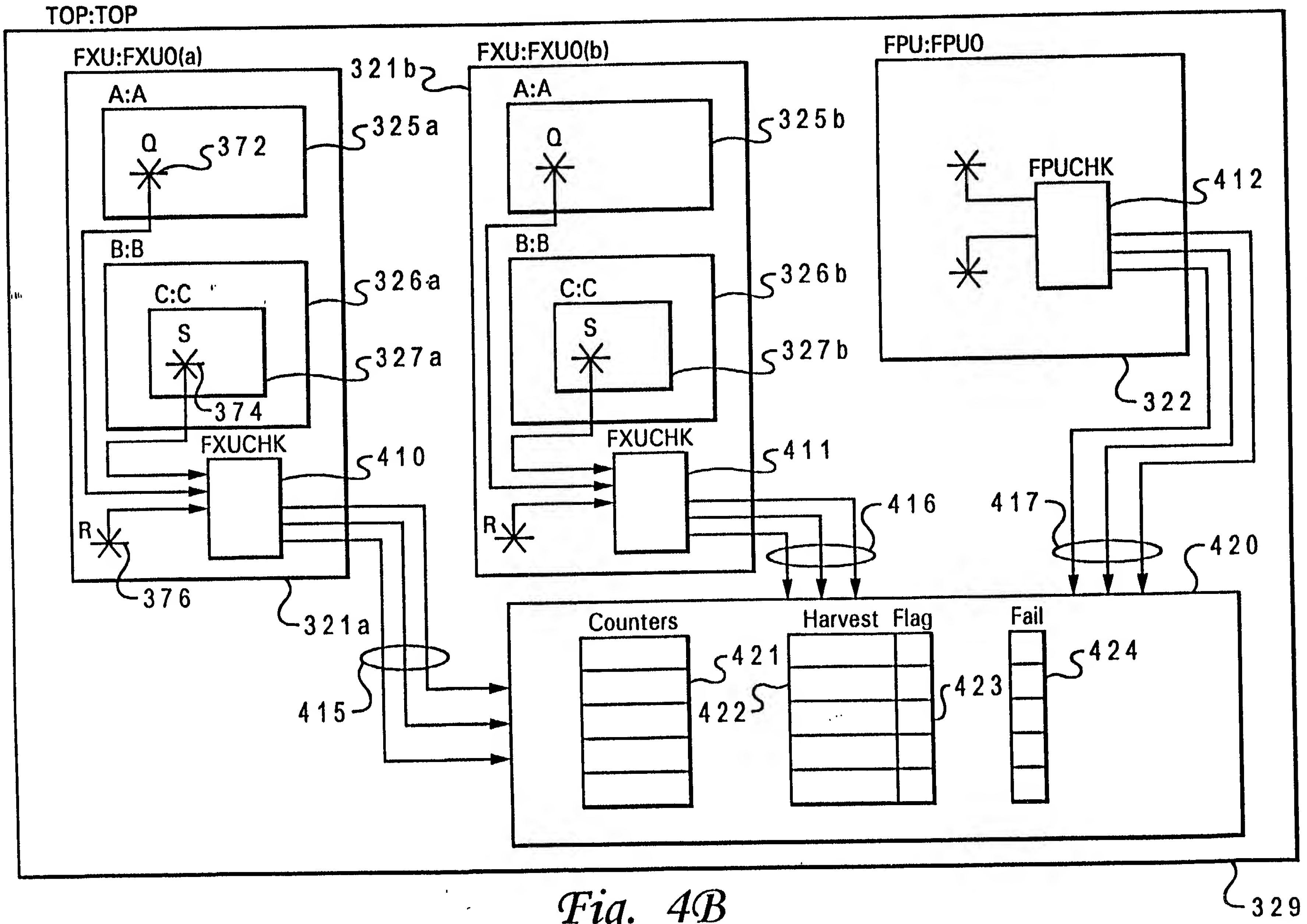
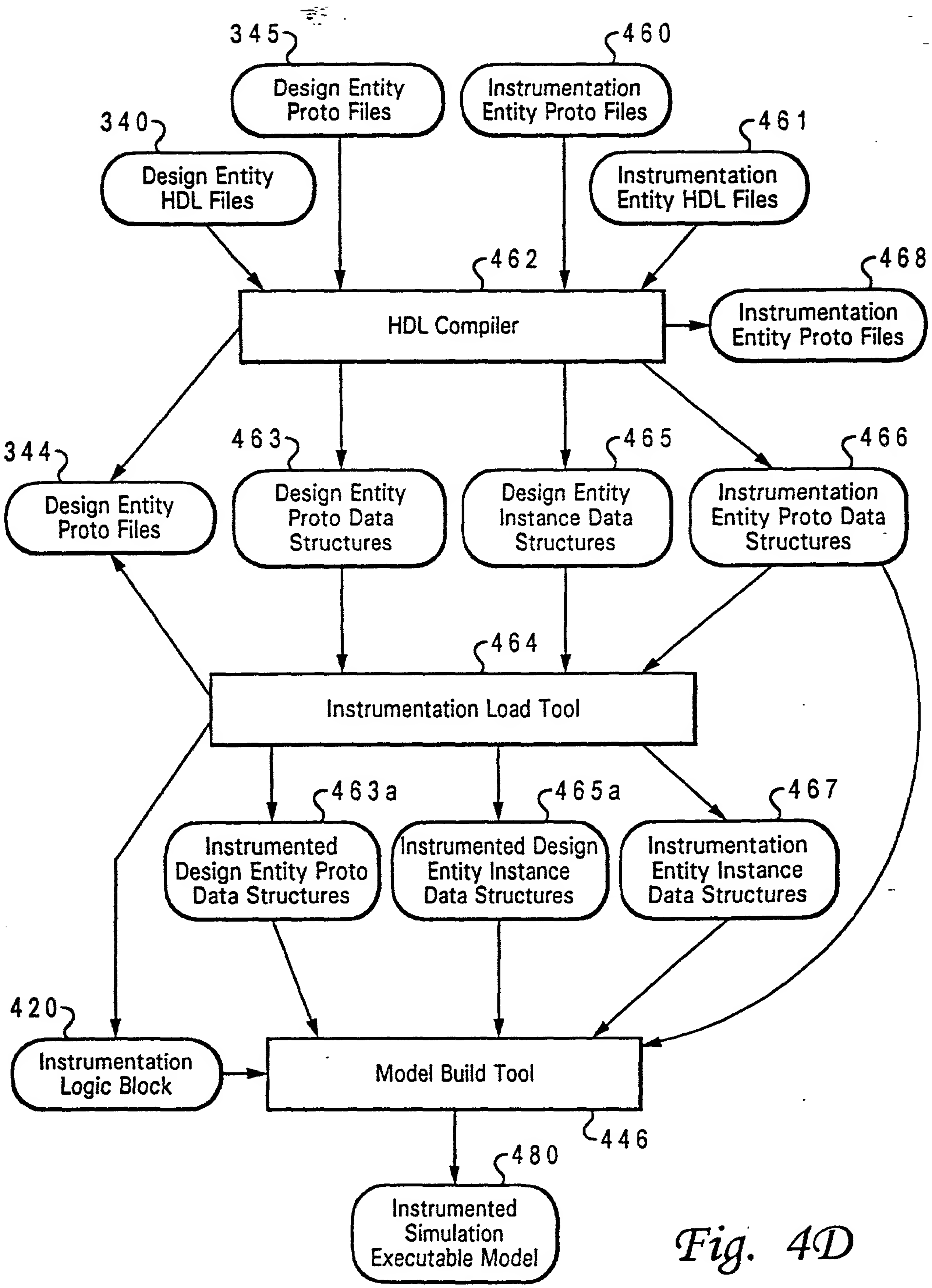

TOP:TOP
FXU:FXU0(a)
321b
FXU:FXU0(b)
FPU:FPU0
A:A
Q
372
325a
B:B
C:C
S
326a
327a
374
FXUCHK
410
R
376
321a
415
A:A
Q
325b
B:B
C:C
S
326b
327b
FXUCHK
411
R
416
FPUCHK
412
322
417
420
Counters
421
422
Harvest
Flag
423
Fail
424
329

Fig. 4B

```
ENTITY FXUCHK IS                                                         } 450  } 440

     PORT(    S_IN        :     IN std_ulogic:
              Q_IN        :     IN std_ulogic:
              R_IN        :     IN std_ulogic:
              clock       :     IN std_ulogic:
              fails       :     OUT std_ulogic_vector(0 to 1);
              counts      :     OUT std_ulogic_vector(0 to 2);
              harvests    :     OUT std_ulogic_vector(0 to 1);
          );

452 { --!! BEGIN                                                          } 451
      --!! Design Entity: FXU;

453 { --!! Inputs
      --!! S_IN        =>      B.C.S;
      --!! Q_IN        =>      A.Q;
      --!! R_IN        =>      R;
      --!! CLOCK       =>      clock;
      --!! End Inputs

454 { --!! Fail Outputs;
      --!! 0 : "Fail message for failure event 0";
      --!! 1 : "Fail message for failure event 1";
      --!! End Fail Outputs;

455 { --!! Count Outputs;
      --!! 0 : <event0> clock;
      --!! 1 : <event1> clock;
      --!! 2 : <event2> clock;
      --!! End Count Outputs;

456 { --!! Harvest Outputs;
      --!! 0 : "Message for harvest event 0";
      --!! 1 : "Message for harvest event 1";
      --!! End Harvest Outputs;

457 { --!! End;

      ARCHITECTURE example of FXUCHK IS

      BEGIN                                                  } 458

           ... HDL code for entity body section ...

      END;
```

Fig. 4C



345
Design Entity Proto Files
460
Instrumentation Entity Proto Files
340
Design Entity HDL Files
461
Instrumentation Entity HDL Files
462
HDL Compiler
468
Instrumentation Entity Proto Files
344
Design Entity Proto Files
463
Design Entity Proto Data Structures
465
Design Entity Instance Data Structures
466
Instrumentation Entity Proto Data Structures
464
Instrumentation Load Tool
463a
Instrumented Design Entity Proto Data Structures
465a
Instrumented Design Entity Instance Data Structures
467
Instrumentation Entity Instance Data Structures
420
Instrumentation Logic Block
Model Build Tool
446
480
Instrumented Simulation Executable Model

Fig. 4D

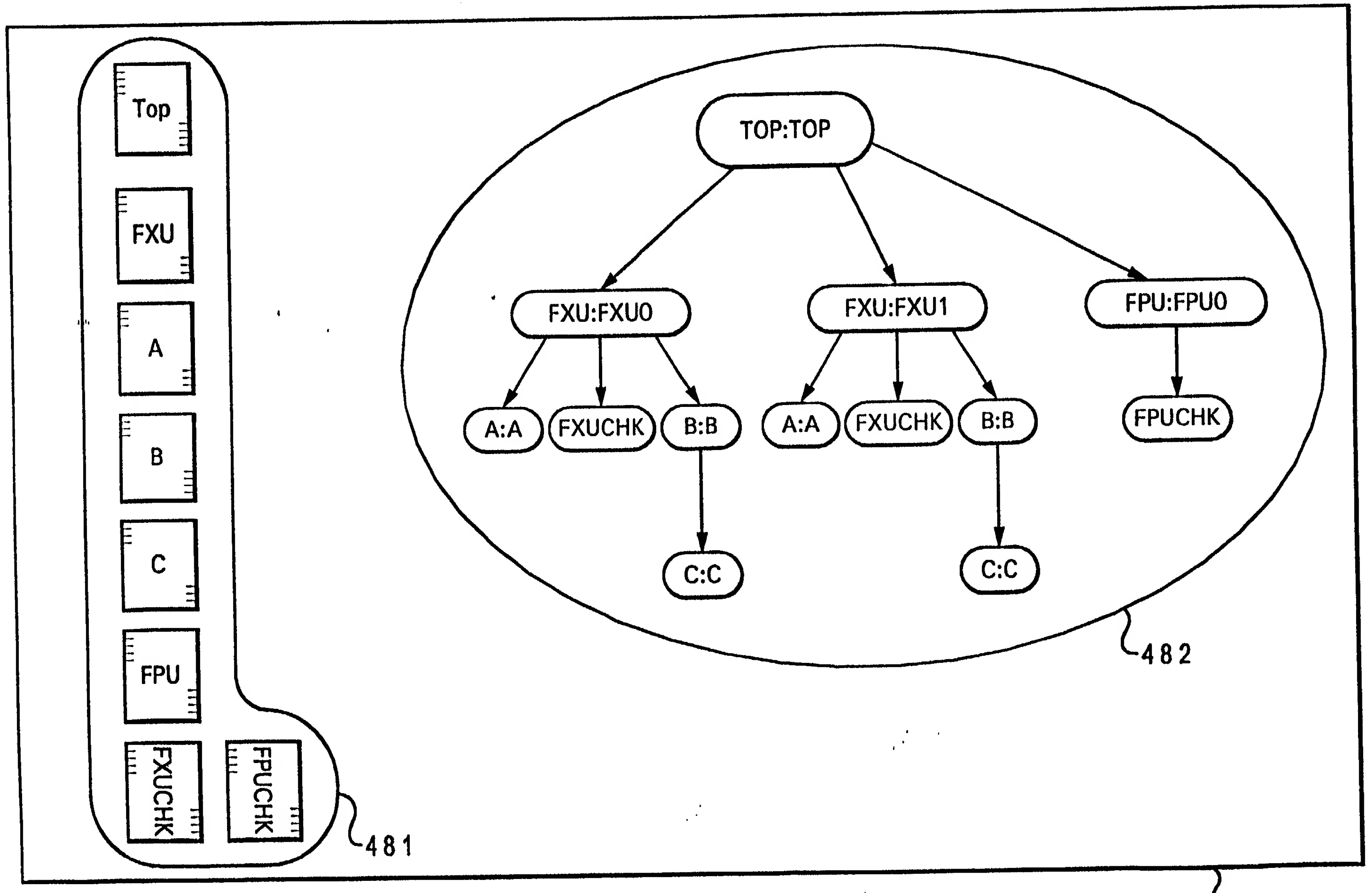

Top
FXU
A
B
C
FPU
FXUCHK
FPUCHK
481
TOP:TOP
FXU:FXU0
FXU:FXU1
FPU:FPU0
A:A
FXUCHK
B:B
A:A
FXUCHK
B:B
FPUCHK
C:C
C:C
482
44

Fig. 4E

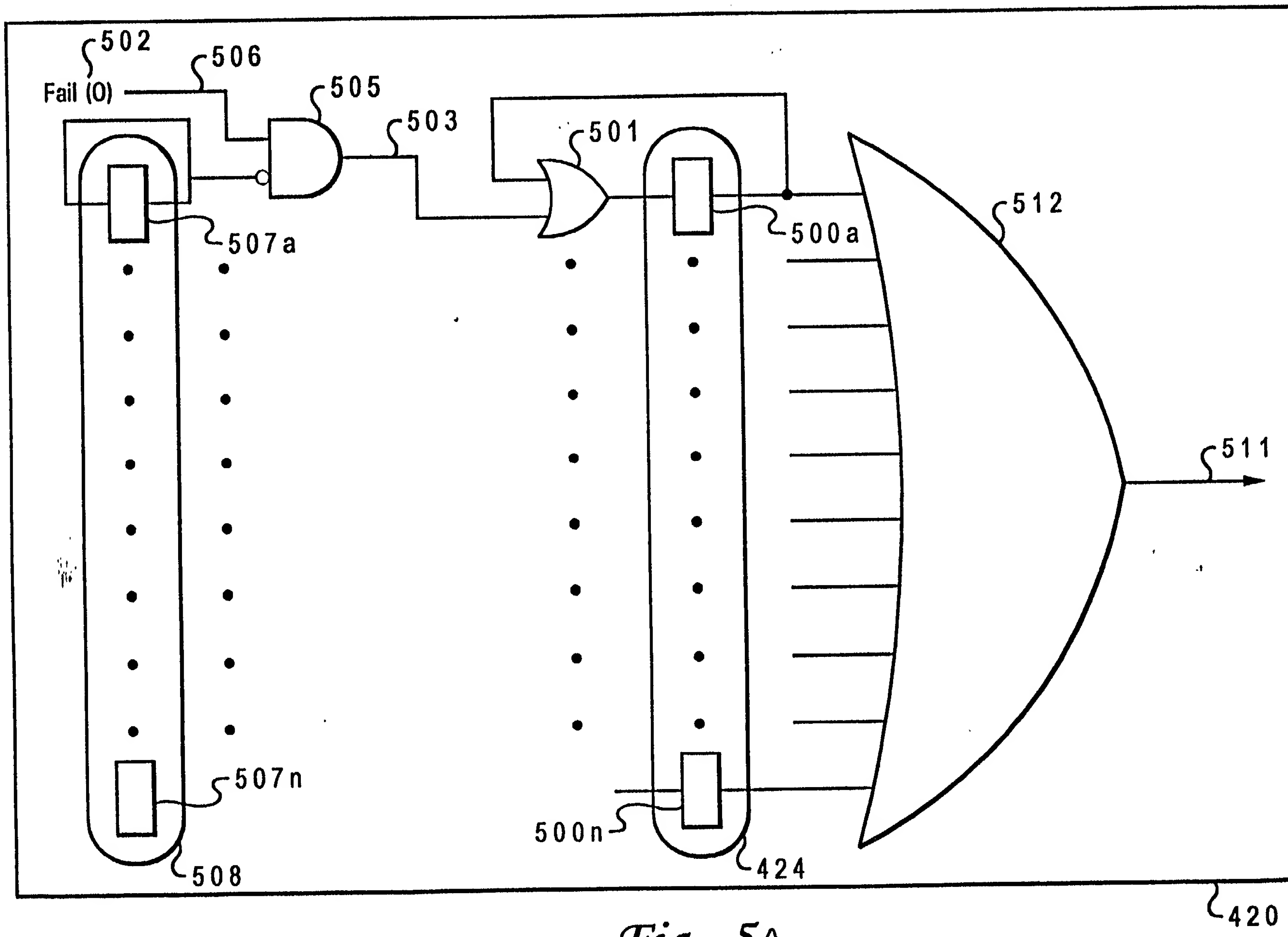

502
Fail (0)
506
505
503
501
507a
500a
512
511
507n
500n
508
424
420

Fig. 5A

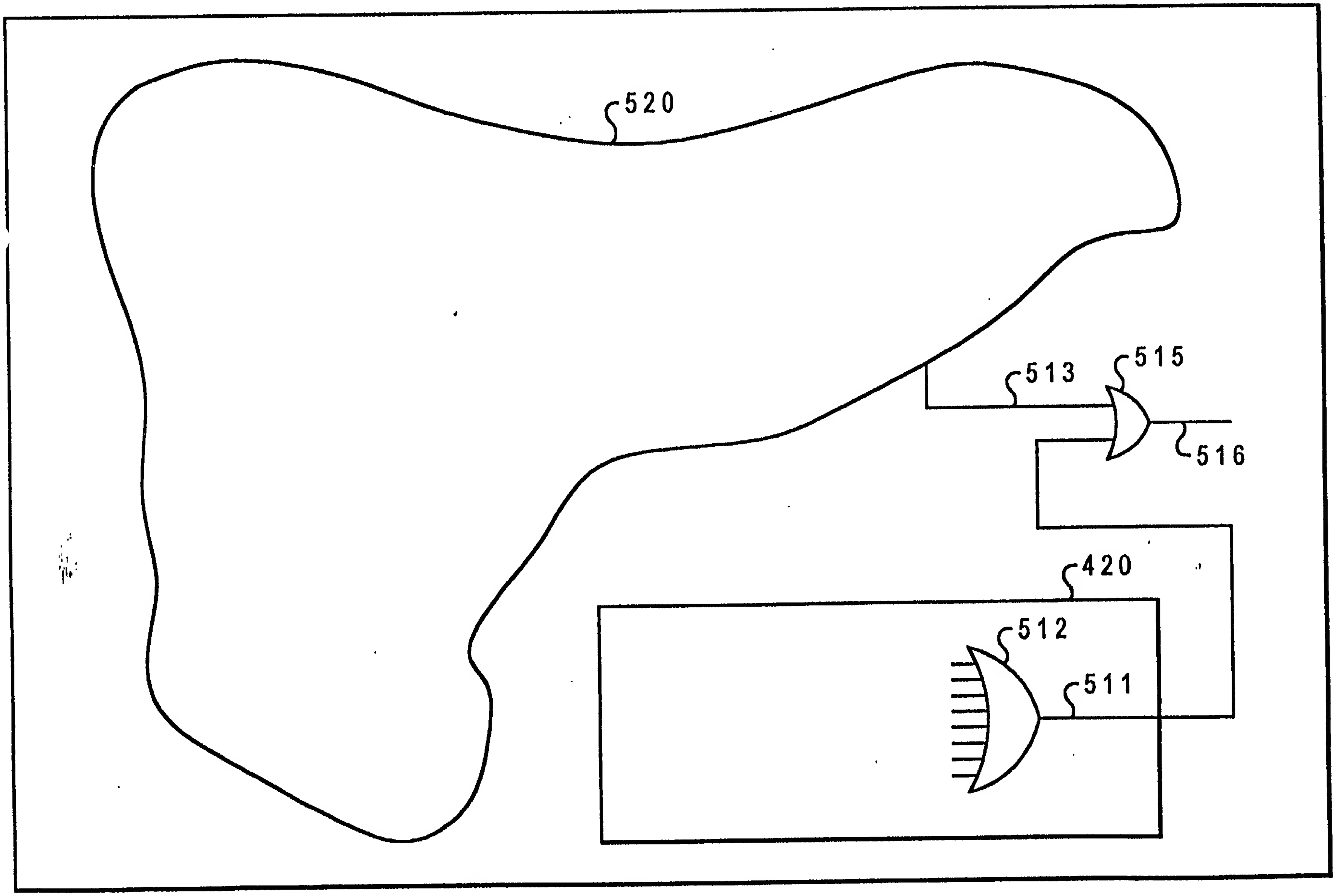


Fig. 5B

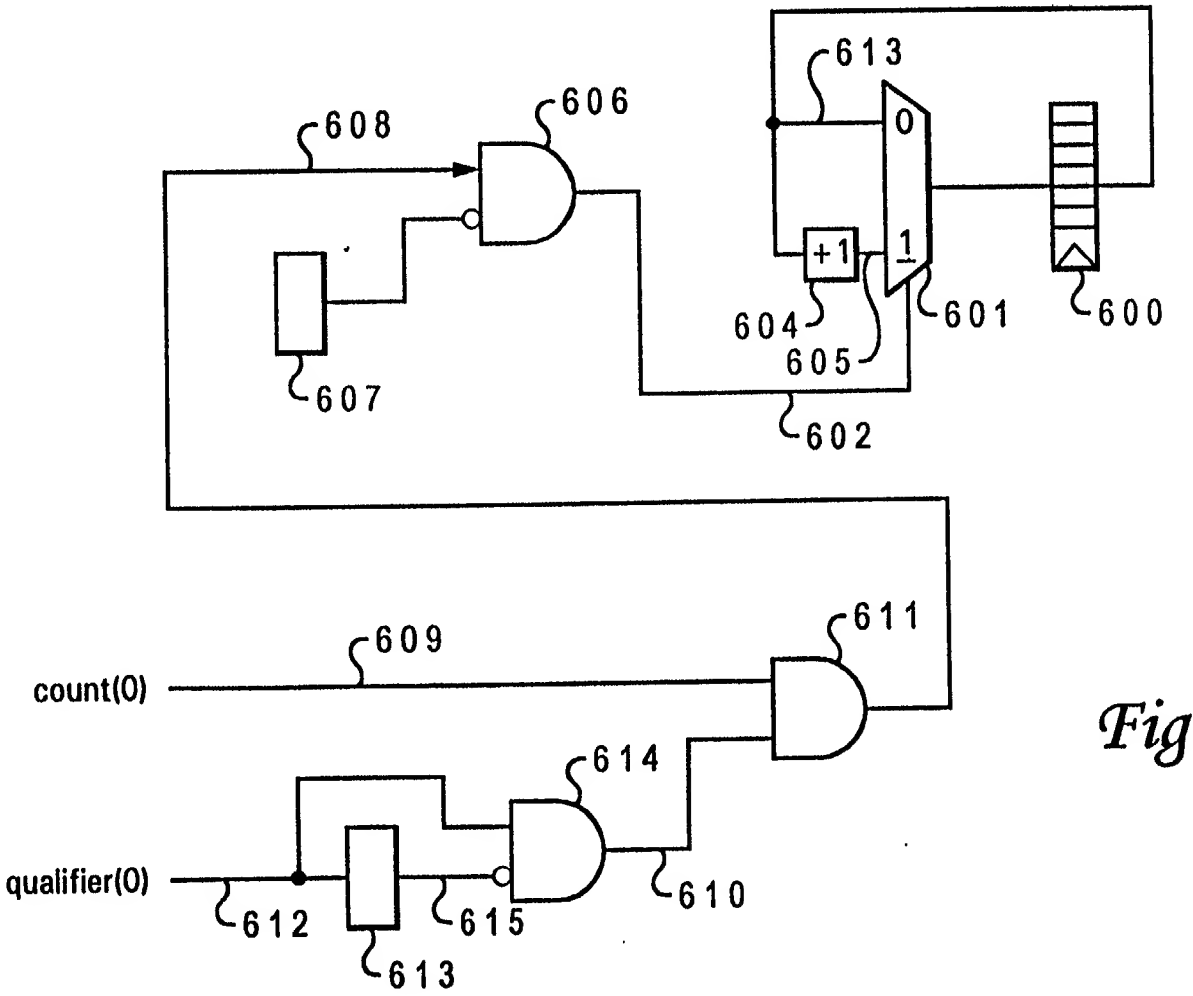

606
608
613
0
1
+1
604
605
601
600
607
602
611
609
count(0)
614
qualifier(0)
612
615
610
613

Fig. 6A

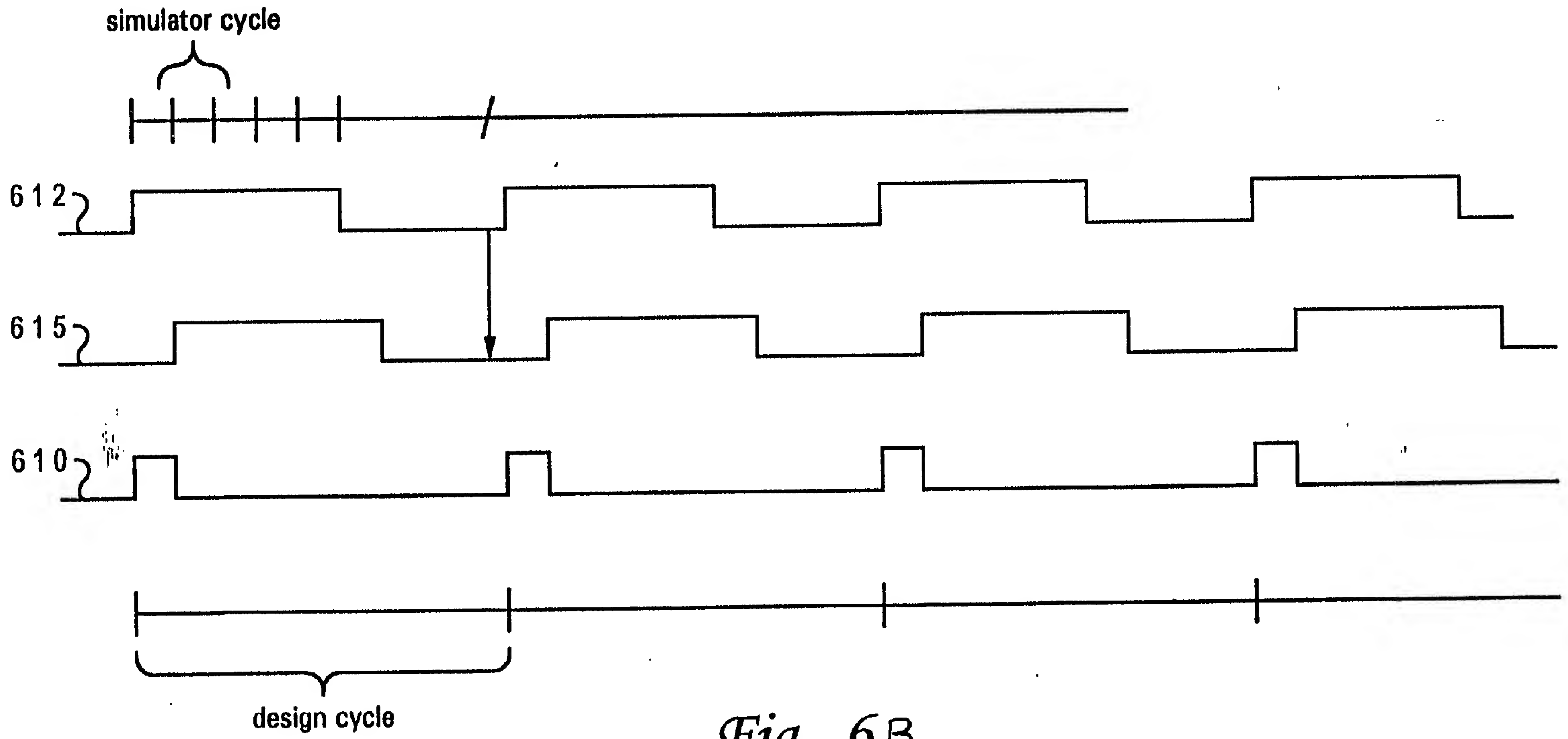


Fig. 6B

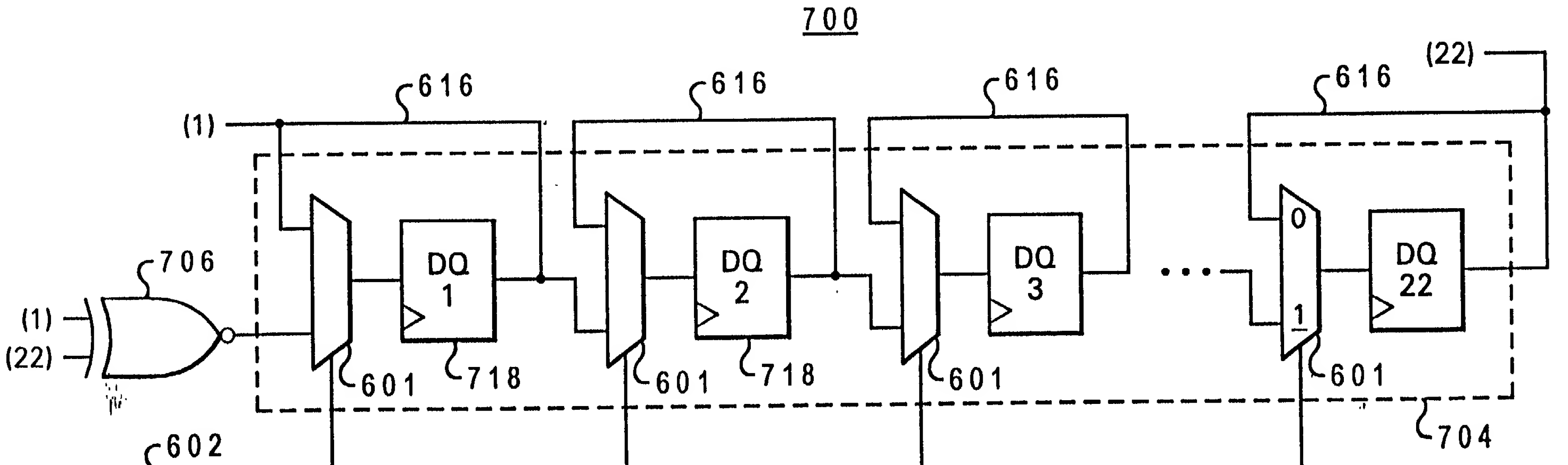

700
(1)
(22)
706
616
616
616
616
(22)
(1)
DQ 1
DQ 2
DQ 3
DQ 22
0
1
601
601
601
601
718
718
704
602

Fig. 7

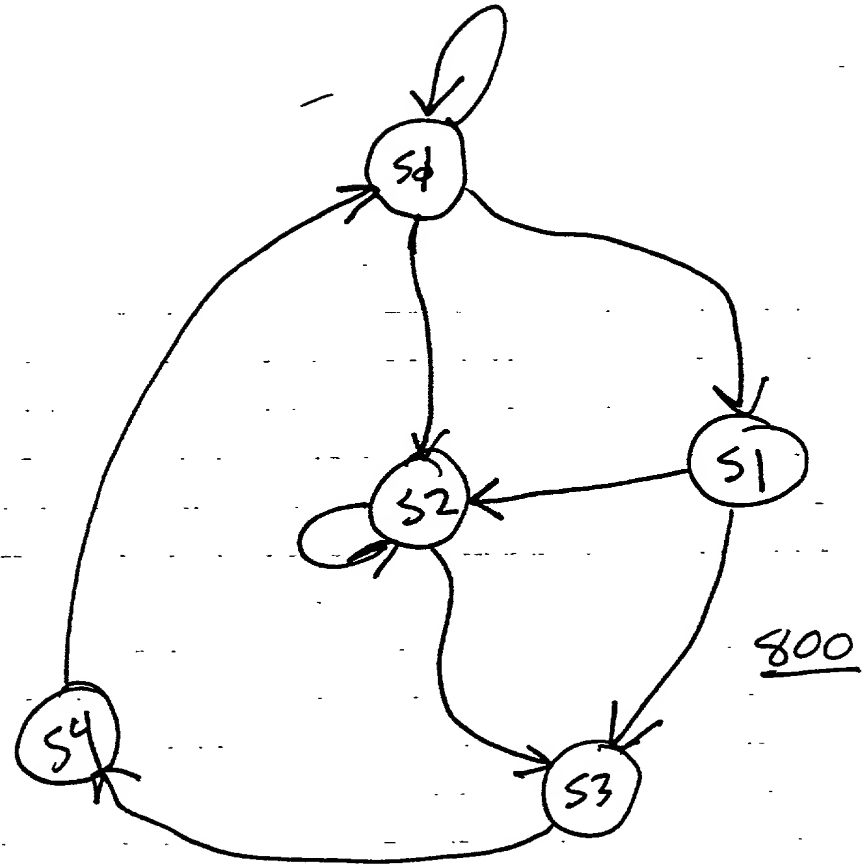


FIG. 8

(Prior Art)

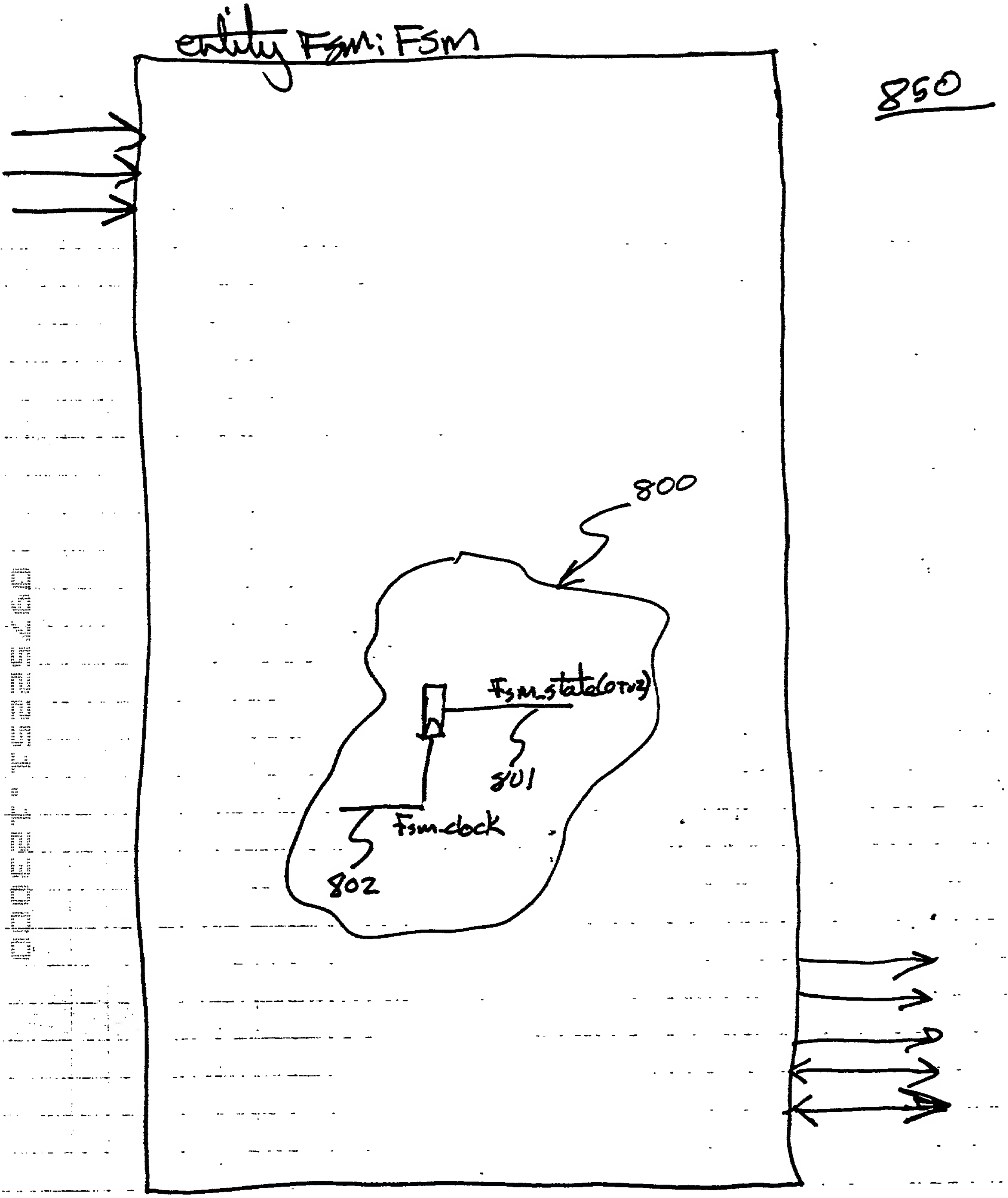


FIG. 8A
(Prior Art)

```
entity Fsm Is
  PORT(
       .... ports for entity Fsm.....
     );

  ARCHITECTURE Fsm of Fsm Is
  BEGIN

      .... HDL code for FSM and rest of the entity...

      fsm-state(0 to 2) <= ...signal 801....

853   --!! Embedded FSM   :exampleFsm;                                        852
859   --!! clock          :(fsm_clock);                                            86
854   --!! state_vector   :(Fsm_state(0 to 2));
855   --!! states         :(s0,s1,s2,s3,s4);
856   --!! state_encoding :('000','001','000','011','100');
857   --!! arcs           :(s0=>s0, s0=>s1, s0=>s2,
      --!!                  s1=>s2, s1=>s3, s2=>s2,
      --!!                  s2=>s3, s3=>s4, s4=>s0);
858   --!! end Fsm;

      .
      .
      .

  END;
```

FIG. 8B

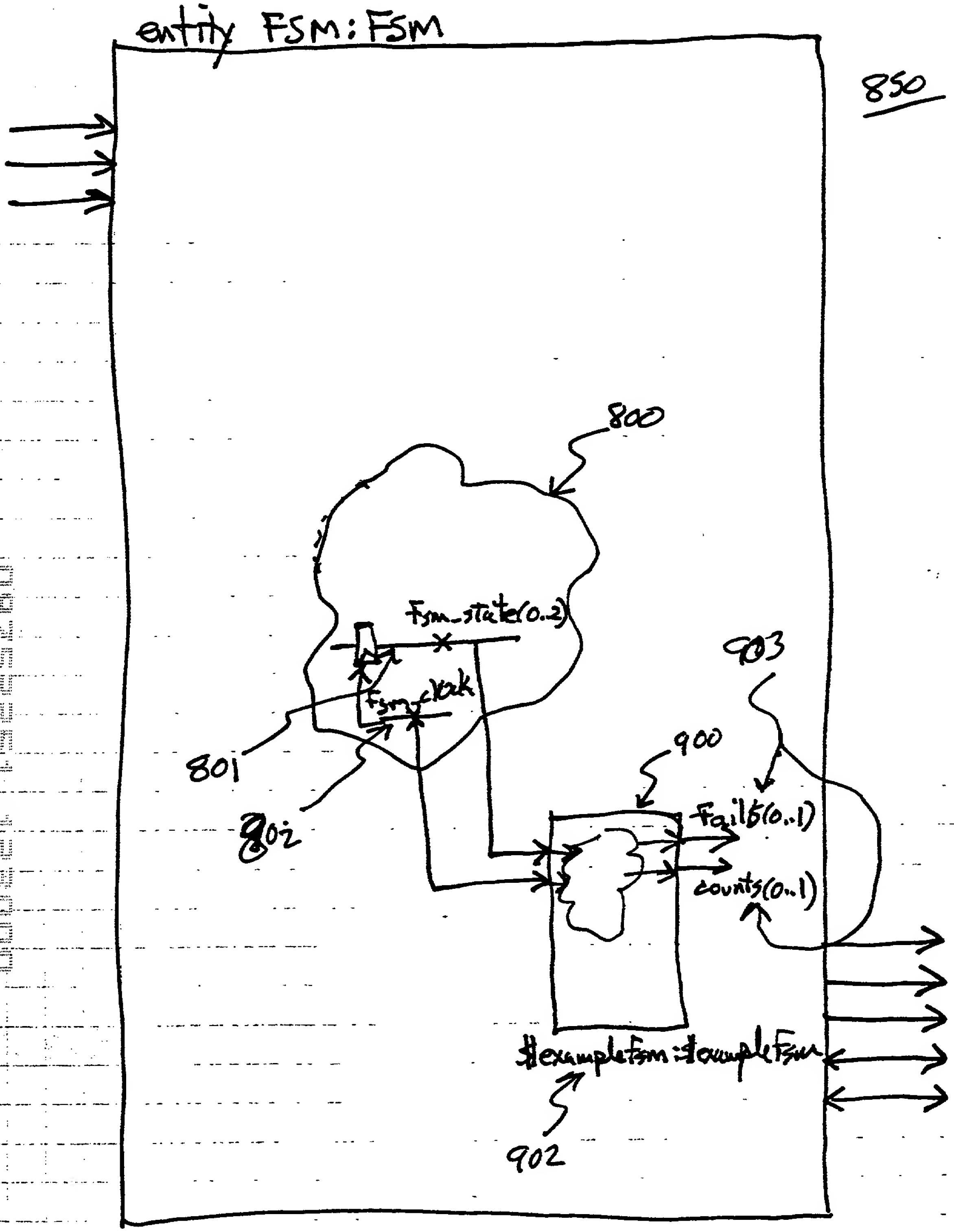

entity FSM:FSM
850
800
Fsm_state(0..2)
Fsm_clock
801
802
903
900
Fails(0..1)
counts(0..1)
$examplefsm:$examplefsm
902

FIG. 9

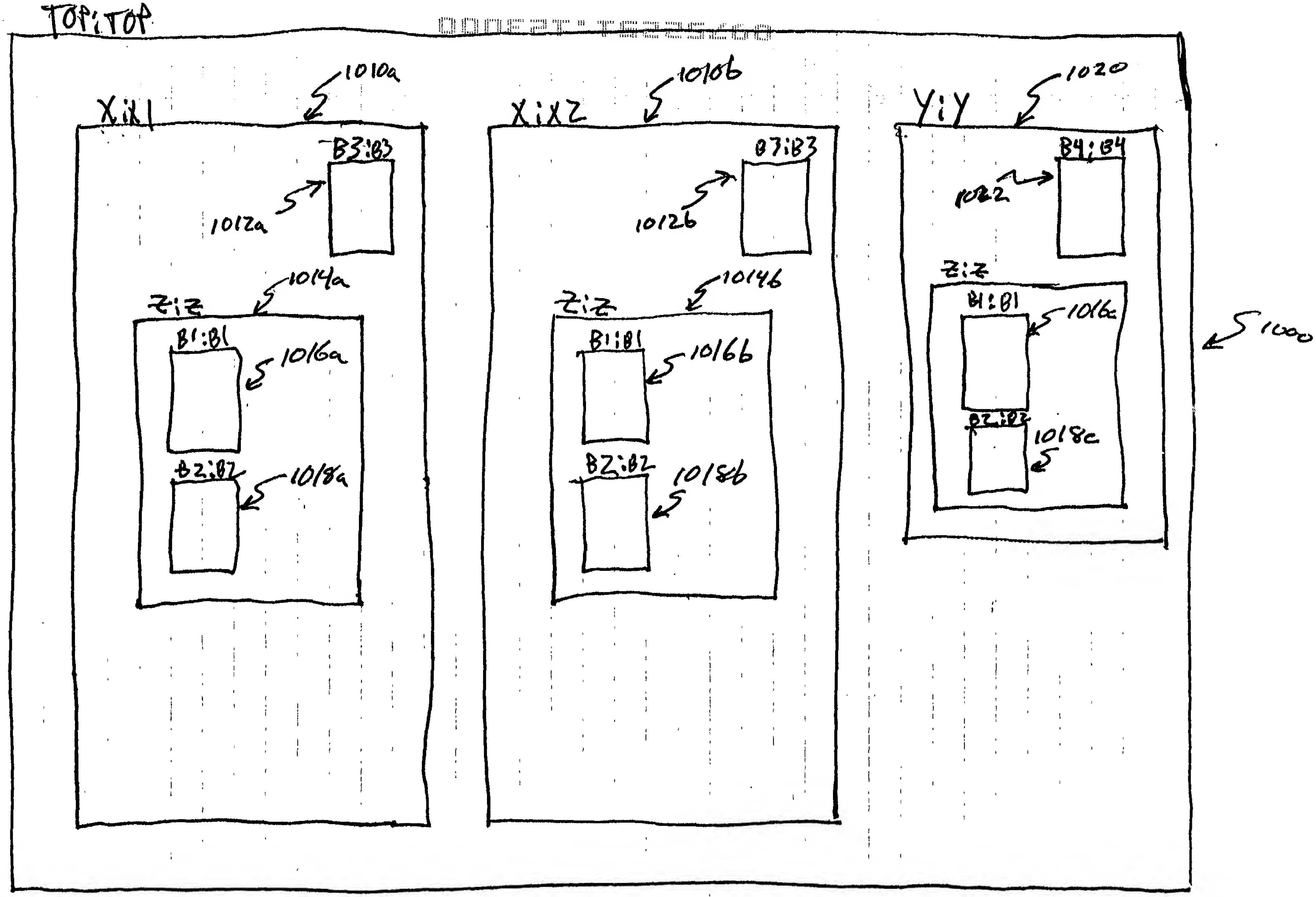

TOP:TOP
1000
X:X1
1010a
B3:B3
1012a
Z:Z
1014a
B1:B1
1016a
B2:B2
1018a
X:X2
1010b
B7:B3
1012b
Z:Z
1014b
B1:B1
1016b
B2:B2
1018b
Y:Y
1020
B4:B4
1022
Z:Z
B1:B1
1016c
B2:B2
1018c

FIG. 10A

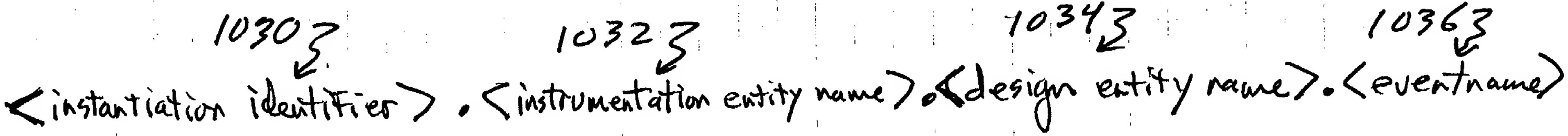
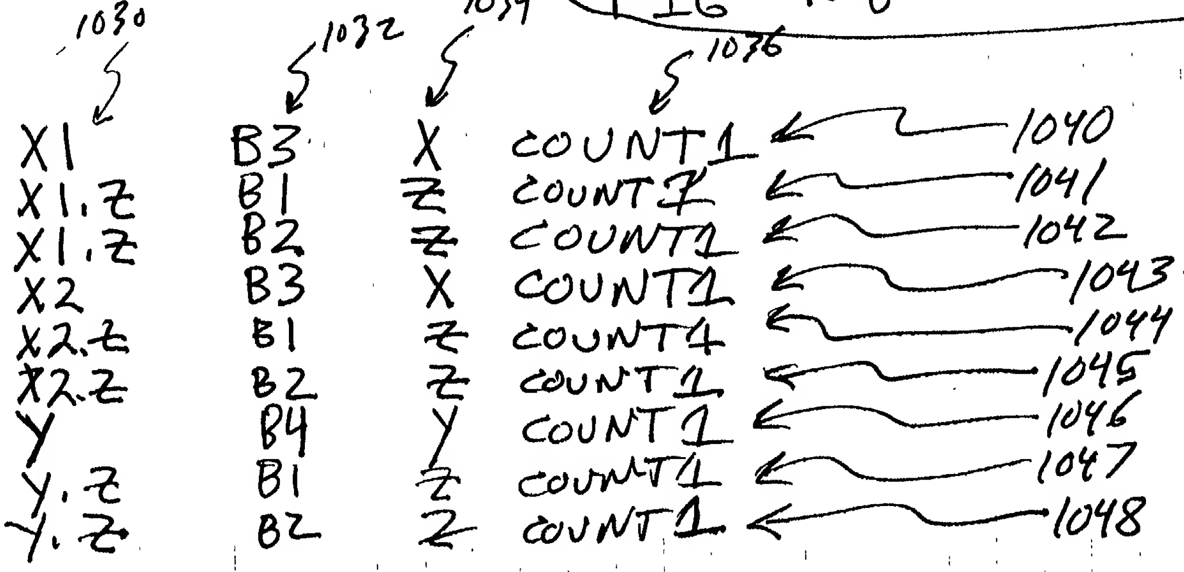


FIG 10B



X1	B3	X	COUNT1
X1.Z	B1	Z	COUNT2
X1.Z	B2	Z	COUNT1
X2	B3	X	COUNT1
X2.Z	B1	Z	COUNT4
X2.Z	B2	Z	COUNT1
Y	B4	Y	COUNT1
Y.Z	B1	Z	COUNT1
Y.Z	B2	Z	COUNT1

FIG 10C

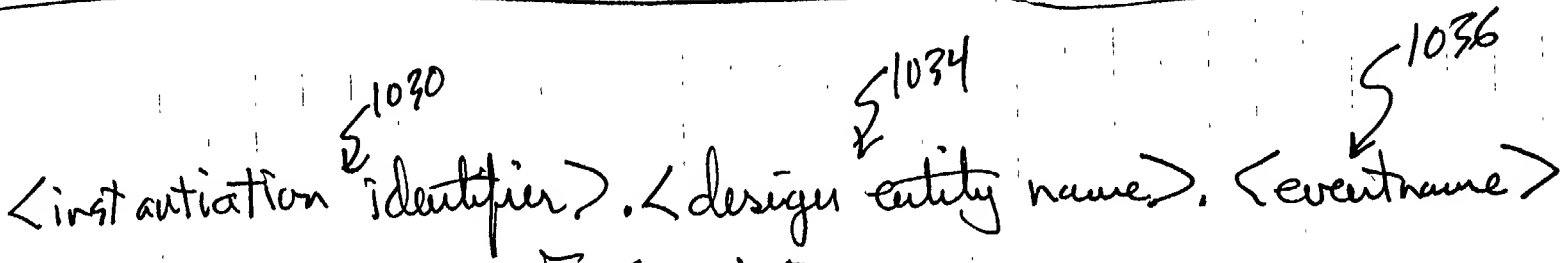


FIG. 10D

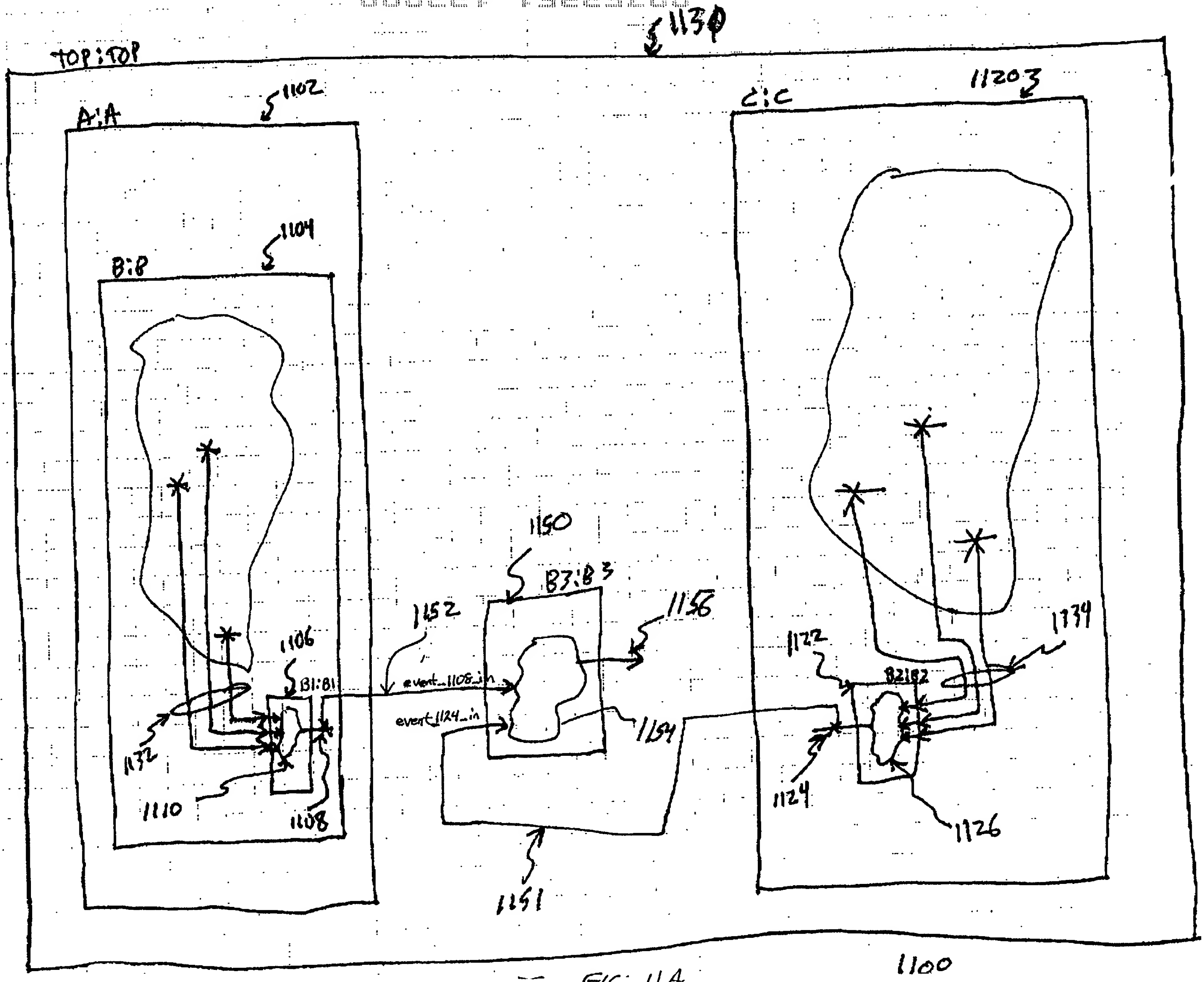
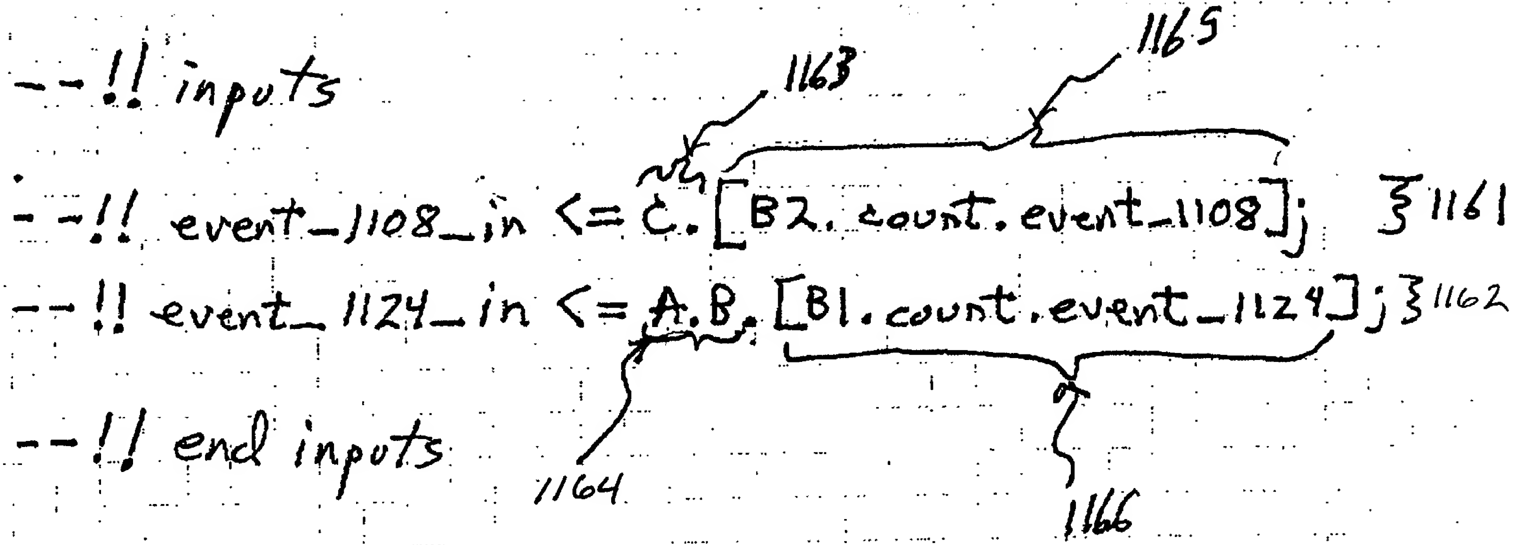
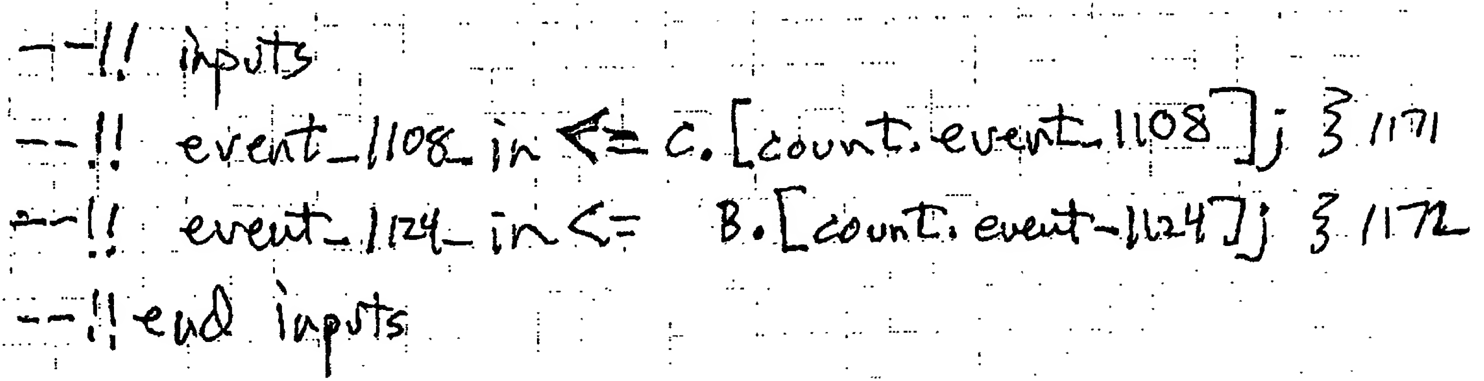

1130
TOP:TOP
A:A
1102
B:B
1104
1106
B1:B1
1132
1110
1108
1152
event_1108_in
event_1124_in
1150
B3:B3
1156
1154
1151
C:C
11203
1122
B2:B2
1134
1124
1126
1100

FIG. 11A



```
--!! inputs
--!! event_1108_in <= C.[B2.count.event_1108];
--!! event_1124_in <= A.B.[B1.count.event_1124];
--!! end inputs
```

FIG. 11B



```
--!! inputs
--!! event_1108_in <= C.[count.event_1108];
--!! event_1124_in <= B.[count.event_1124];
--!! end inputs
```

FIG. 11C

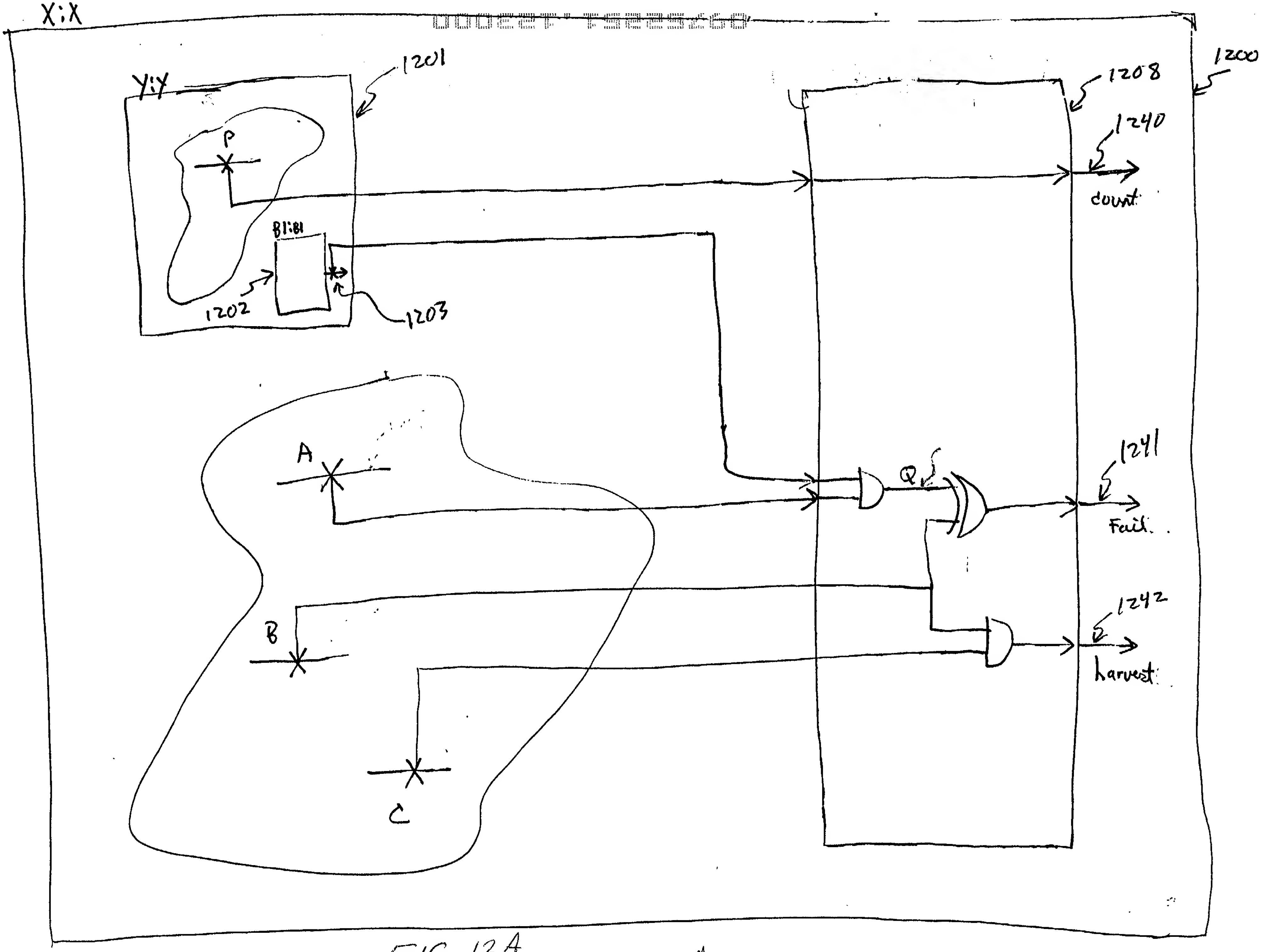

X:X
Y:Y
1201
P
B1:B1
1202
1203
1208
1200
1240
count
A
Q
1241
Fail
B
1242
harvest
C

FIG. 12A

```
Entity X IS
    PORT ( ;
           :
         );

ARCHITECTURE example OF X IS

BEGIN
      :
    ...HDL CODE FOR X.----
      :

    Y: Y                    }
    PORT MAP ( ;            } 1221
               );           }

    A <= ...                }
    B <= ...                } 1222
    C <= ...                }

    --!! [count, countname0, clock] <= Y.P;                 } 1230  }
    --!! Q <= Y.[B1.count.count1] AND A;                     } 1232  }
    --!! [fail, Failname0, "fail msg"] <= Q XOR B;           } 1234  } 1223
    --!! [harvest, harvestname0, "harvest msg"] <= B AND C;  } 1236  }

END
```

122

FIG. 12B